

An algorithm for discrete fractional Hadamard transform

Aleksandr Cariow · Dorota
Majorkowska-Mech

Received: date / Accepted: date

Abstract We present a novel algorithm for calculating the discrete fractional Hadamard transform for data vectors whose size N is a power of two. A direct method for calculation of the discrete fractional Hadamard transform requires N^2 multiplications, while in proposed algorithm the number of real multiplications is reduced to $N\log_2 N$.

Keywords Discrete linear transforms · Discrete fractional Hadamard transform · Eigenvalue decomposition · Fast algorithms

Mathematics Subject Classification (2000) MSC 65Y20 · MSC 15A04 · MSC 15A18

1 Introduction

Discrete fractional transforms are the generalizations of the ordinary discrete transforms with one additional fractional parameter. In the past decades, various discrete fractional transforms including discrete Fourier transform [1], [2], discrete fractional Hartley transform [3], discrete fractional cosine transforms

A. Cariow
Faculty of Computer Science and Information Technology, West Pomeranian University of Technology, Szczecin, Poland
Tel.: +48-91-4495573
Fax: +48-91-4495559
E-mail: atariow@wi.zut.edu.pl

D. Majorkowska-Mech
Faculty of Computer Science and Information Technology, West Pomeranian University of Technology, Szczecin, Poland
Tel.: +48-91-4495582
Fax: +48-91-4495559
E-mail: dmajorkowska@wi.zut.edu.pl

and discrete sine transform [4] have been introduced and found wide applications in many scientific and technological areas including digital signal processing [5], image encryption [6], [7], [8] and digital watermarking [9] and others. Different fast algorithms for their implementations have been separately developed to minimize computational complexity and implementation costs. In [10] a discrete fractional Hadamard transform for the vector of length $N = 2^n$ was introduced, however a fast algorithm for the realization of this transform has not been proposed. In our previous paper [11] we describe a rationalized algorithm for DFRHT possessing a reduced number of multiplications and additions. Analysis of the mentioned algorithm shows that not all of existing improvement possibilities have been realized. In this paper, we proposed a novel algorithm for the discrete fractional Hadamard transform that require fewer total real additions and multiplications than our previously published solution.

2 Mathematical background

A Hadamard matrix of order N is a $N \times N$ symmetric matrix whose entries are either 1 or -1 and whose rows are mutually orthogonal. In this paper we will use the normalized form of this matrix and we will denote it by \mathbf{H}_N . For $N = 2^n$ the Hadamard matrices can be recursively obtained due to Sylvester's construction [12]:

$$\mathbf{H}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad \mathbf{H}_N = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{H}_{\frac{N}{2}} & \mathbf{H}_{\frac{N}{2}} \\ \mathbf{H}_{\frac{N}{2}} & -\mathbf{H}_{\frac{N}{2}} \end{bmatrix} \quad (1)$$

for $N = 4, 8, \dots, 2^n$.

Definition of the discrete fractional Hadamard (DFRHT) transform is based on an eigenvalue decomposition of the DHT matrix. Any real symmetric matrix (including the Hadamard matrix) can be diagonalized, e.g. written as a product [13]

$$\mathbf{H}_N = \mathbf{Z}_N \mathbf{\Lambda}_N \mathbf{Z}_N^T = \sum_{k=0}^{N-1} \lambda_k \mathbf{z}_N^{(k)} (\mathbf{z}_N^{(k)})^T \quad (2)$$

where $\mathbf{\Lambda}_N$ is a diagonal matrix of order 2^n , whose diagonal entries are the eigenvalues of \mathbf{H}_N

$$\mathbf{\Lambda}_N = \begin{bmatrix} \lambda_0 & & & \\ & \lambda_1 & & \mathbf{0} \\ & & \ddots & \\ \mathbf{0} & & & \lambda_{N-1} \end{bmatrix} \quad (3)$$

$\mathbf{Z}_N = [\mathbf{z}_N^{(0)} \mid \mathbf{z}_N^{(1)} \mid \dots \mid \mathbf{z}_N^{(N-1)}]$ - the matrix whose columns are normalized

mutually orthogonal eigenvectors of the matrix \mathbf{H}_N . The eigenvector $\mathbf{z}_N^{(k)}$ is related to the eigenvalue λ_k . A superscript T denotes the matrix transposition.

The DFRHT matrix of order $N = 2^n$ with real parameter α was first defined in [10]. This matrix can be regarded as a power of the DHT matrix, where the exponent $a = \alpha/\pi$

$$\mathbf{H}_N^a = \mathbf{Z}_N \mathbf{\Lambda}_N^a \mathbf{Z}_N^T = \sum_{k=0}^{N-1} \lambda_k^a \mathbf{z}_N^{(k)} (\mathbf{z}_N^{(k)})^T \quad (4)$$

For $a = 0$ the DFRHT matrix is converted into the identity matrix, and for $a=1$ it is transformed into the ordinary DHT matrix. Generally the DFRHT matrix is complex-valued.

An essential operation, by obtaining the discrete fractional Hadamard matrix, defined by (4), is calculating the eigenvalues and the eigenvectors of the matrix \mathbf{H}_N . The only eigenvalues of the unnormalized Hadamard matrix of order $N = 2^n$ are known to be $2^{n/2}$ and $-2^{n/2}$ [14], hence the normalized Hadamard matrix \mathbf{H}_N has only the eigenvalues 1 and -1 . A method for finding the eigenvectors of Hadamard matrix was firstly presented in [15], but in [10] a recursive method for calculation the eigenvectors of the Hadamard matrix order 2^{n+1} based on the eigenvectors of the Hadamard matrix of order 2^n has been proposed. We will use this method to obtain the DFRHT matrix. Here we will present it briefly.

In [15] it was proven that if $\mathbf{v}_N^{(k)}$ ($k=0, 1, \dots, N-1$) is an eigenvector of Hadamard matrix of order $N=2^n$ associated with an eigenvalue λ , then vector

$$\hat{\mathbf{v}}_{2N}^{(k)} = \begin{bmatrix} \mathbf{v}_N^{(k)} \\ (\sqrt{2}-1)\mathbf{v}_N^{(k)} \end{bmatrix} \quad (5)$$

will be an eigenvector of the matrix \mathbf{H}_{2N} associated with the eigenvalue λ .

In [10] it was proven that if $\mathbf{v}_N^{(k)}$ is an eigenvector of \mathbf{H}_N associated with an eigenvalue λ , then the vector

$$\tilde{\mathbf{v}}_{2N}^{(k)} = \begin{bmatrix} (1-\sqrt{2})\mathbf{v}_N^{(k)} \\ \mathbf{v}_N^{(k)} \end{bmatrix} \quad (6)$$

will be an eigenvector of the matrix \mathbf{H}_{2N} associated with the eigenvalue $-\lambda$. These two results allow us to generate the eigenvectors of Hadamard matrix of order 2^{n+1} from the eigenvectors of Hadamard Matrix of order 2^n . Knowing the straightforward calculated eigenvectors of the matrix \mathbf{H}_2

$$\mathbf{v}_2^{(0)} = \begin{bmatrix} 1 \\ \sqrt{2}-1 \end{bmatrix} \quad \mathbf{v}_2^{(1)} = \begin{bmatrix} 1-\sqrt{2} \\ 1 \end{bmatrix} \quad (7)$$

associated with eigenvalues 1 and -1 respectively, the eigenvectors for Hadamard matrix of arbitrary order $N=2^n$ can be recursively computed. In [10] it was also shown so this recursively computed eigenvectors of matrix \mathbf{H}_N will be orthogonal. It should be noted that for any $N=2^n$ there are only two distinct

eigenvalues of Hadamard matrix, so for $N \geq 4$ the eigenvalues are degenerated. Because of this fact the set of eigenvectors proposed in [15] and [10] is not unique. The eigenvectors $\mathbf{v}_N^{(k)}$ for $k = 0, 1, \dots, N-1$, which are columns of the matrix \mathbf{Z}_N (after normalization), as well as their associated eigenvalues λ_k , can be however ordered in different ways. In [10] it has been also established a method of ordering the eigenvectors. In many cases, including the case of discrete fractional transforms is used so-called sequency ordering of the eigenvectors. This means that the k -th eigenvector has k sign-changes. The discrete Hermite-Gaussians, eigenvectors of discrete Fourier transform matrix are ordered this way as well [2]. We will show this method of ordering of the eigenvectors in Example 1.

Example 1 The number of sign-changes in eigenvectors $\mathbf{v}_2^{(0)}$ and $\mathbf{v}_2^{(1)}$ of matrix \mathbf{H}_2 , determined by (7), is equal to 0 and 1 respectively. Using expressions (5) and (6) we obtain the eigenvectors of matrix \mathbf{H}_4 :

$$\hat{\mathbf{v}}_4^{(0)} = \begin{bmatrix} 1 \\ b \\ b \\ b^2 \end{bmatrix} \quad \tilde{\mathbf{v}}_4^{(0)} = \begin{bmatrix} -b \\ -b^2 \\ 1 \\ b \end{bmatrix} \quad \hat{\mathbf{v}}_4^{(1)} = \begin{bmatrix} -b \\ 1 \\ -b^2 \\ b \end{bmatrix} \quad \tilde{\mathbf{v}}_4^{(1)} = \begin{bmatrix} b^2 \\ -b \\ -b \\ 1 \end{bmatrix},$$

where $b = \sqrt{2} - 1$. The numbers of sign-changes in the above vectors are 0, 1, 3, 2 respectively ($b > 0$). Therefore, a sequency ordered set of eigenvectors of matrix \mathbf{H}_4 will be as follows:

$$\mathbf{v}_4^{(0)} = \hat{\mathbf{v}}_4^{(0)} \quad \mathbf{v}_4^{(1)} = \tilde{\mathbf{v}}_4^{(0)} \quad \mathbf{v}_4^{(2)} = \tilde{\mathbf{v}}_4^{(1)} \quad \mathbf{v}_4^{(3)} = \hat{\mathbf{v}}_4^{(1)}.$$

The corresponding eigenvalues will be equal to:

$$\lambda_0 = 1 \quad \lambda_1 = -1 \quad \lambda_2 = 1 \quad \lambda_3 = -1.$$

The relations obtained in Example 1 can be easily generalized as follows:

$$\begin{cases} \mathbf{v}_{2N}^{(4l)} = \hat{\mathbf{v}}_{2N}^{(2l)} \\ \mathbf{v}_{2N}^{(4l+1)} = \tilde{\mathbf{v}}_{2N}^{(2l)} \\ \mathbf{v}_{2N}^{(4l+2)} = \tilde{\mathbf{v}}_{2N}^{(2l+1)} \\ \mathbf{v}_{2N}^{(4l+3)} = \hat{\mathbf{v}}_{2N}^{(2l+1)} \end{cases} \quad (8)$$

for $l = 0, 1, \dots, \frac{N}{2} - 1$ and

$$\lambda_k = (-1)^k. \quad (9)$$

for $k = 0, 1, \dots, 2N - 1$.

Both the eigenvectors of the matrix \mathbf{H}_2 and the eigenvectors obtained for higher order Hadamard matrices are not normalized. Let the notation $\|\mathbf{v}_N^{(k)}\|$

means the Euclidean norm of vector $\mathbf{v}_N^{(k)}$. In [11] it was shown that for any $N = 2^n$ we have the relationship

$$\|\mathbf{v}_N^{(k)}\|^2 = (1 + b^2)^n \quad (10)$$

where $k = 0, 1, \dots, N-1$ and $b = \sqrt{2} - 1$. If we take the designation $c = 1 + b^2$, then normalized eigenvectors of Hadamard matrix of order $N = 2^n$ will take the form

$$\mathbf{z}_N^{(k)} = \frac{\mathbf{v}_N^{(k)}}{\|\mathbf{v}_N^{(k)}\|} = \frac{\mathbf{v}_N^{(k)}}{\sqrt{c^n}} \quad (11)$$

Using the normalized and sequency ordered eigenvectors of the Hadamard matrix, the eigenvalue decomposition (2) of the Hadamard matrix can be written as follows:

$$\mathbf{H}_N = \mathbf{Z}_N \mathbf{\Lambda}_N \mathbf{Z}_N^T = \frac{1}{c^n} \mathbf{V}_N \mathbf{\Lambda}_N \mathbf{V}_N^T \quad (12)$$

where $\mathbf{\Lambda}_N$ is the diagonal matrix whose non-zero elements are

$$\lambda_k = (-1)^k = e^{-jk\pi} \quad (13)$$

for $k = 0, 1, \dots, N-1$. Hence the definition (4) of DFRHT matrix will take the form:

$$\mathbf{H}_N^a = \frac{1}{c^n} \mathbf{V}_N \mathbf{\Lambda}_N^a \mathbf{V}_N^T \quad (14)$$

where

$$\lambda_k^a = e^{-jk\pi a} \quad (15)$$

for $k = 0, 1, \dots, N-1$.

Our goal is to calculate the discrete fractional Hadamard transform for an input signal \mathbf{x}_N in which the number of samples is equal to $N = 2^n$. By $\mathbf{y}_N^{(a)}$ we denote an output signal which is calculated using the formula

$$\mathbf{y}_N^{(a)} = \mathbf{H}_N^a \mathbf{x}_N \quad (16)$$

Supposing that the matrix \mathbf{H}_N^a is given, to calculate the output signal it is necessary to perform N^2 complex multiplications and $N(N-1)$ complex additions. If the input signal is real, then the number of real multiplications will be equal to $2N^2$, and the number of real additions will be equal to $2N(N-1)$.

If we use the decomposition (14) of the matrix \mathbf{H}_N^a by calculating (16) and will perform the matrix-vector multiplication from the right side to the left, the most time-consuming operations are multiplications of matrices \mathbf{V}_N^T and \mathbf{V}_N by the vector, because those matrices are not diagonal. If we interchange the columns of the matrix \mathbf{V}_N in the prescribed manner, we obtain a matrix $\bar{\mathbf{V}}_N$ of special structure, which can be generated recursively. We will show it in Example 2. It will allow to reduce the number of arithmetical operations by calculating the products of matrices $\bar{\mathbf{V}}_N^T$ and $\bar{\mathbf{V}}_N$ by the vector.

Example 2 The matrices \mathbf{V}_N for $N = 2, 4, 8$ are as follows:

$$\mathbf{V}_2 = \begin{bmatrix} 1 & -b \\ b & 1 \end{bmatrix}, \quad \mathbf{V}_4 = \begin{bmatrix} 1 & -b & b^2 & -b \\ b & -b^2 & -b & 1 \\ b & 1 & -b & -b^2 \\ b^2 & b & 1 & b \end{bmatrix},$$

$$\mathbf{V}_8 = \begin{bmatrix} 1 & -b & b^2 & -b & b^2 & -b^3 & b^2 & -b \\ b & -b^2 & b^3 & -b^2 & -b & b^2 & -b & 1 \\ b & -b^2 & -b & 1 & -b & b^2 & b^3 & -b^2 \\ b^2 & -b^3 & -b^2 & b & 1 & -b & -b^2 & b \\ b & 1 & -b & -b^2 & b^3 & b^2 & -b & -b^2 \\ b^2 & b & -b^2 & -b^3 & -b^2 & -b & 1 & b \\ b^2 & b & 1 & b & -b^2 & -b & -b^2 & -b^3 \\ b^3 & b^2 & b & b^2 & b & 1 & b & b^2 \end{bmatrix}.$$

The matrix \mathbf{V}_2 has some specific structure. Now we consider the matrix \mathbf{V}_4 . If in the matrix \mathbf{V}_4 the second and fourth columns will be interchange and then the third and fourth columns will be interchange too, we obtain the following matrix:

$$\bar{\mathbf{V}}_4 = \begin{bmatrix} 1 & -b & -b & b^2 \\ b & 1 & -b^2 & -b \\ b & -b^2 & 1 & -b \\ b^2 & b & b & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{V}_2 & -b\mathbf{V}_2 \\ b\mathbf{V}_2 & \mathbf{V}_2 \end{bmatrix}.$$

The matrix \mathbf{V}_4 differs from the matrix $\bar{\mathbf{V}}_4$ only in the order of the columns. Therefore, the matrix \mathbf{V}_4 can be obtained by post-multiplying the matrix $\bar{\mathbf{V}}_4$ by the permutation matrix \mathbf{P}_4 :

$$\mathbf{V}_4 = \bar{\mathbf{V}}_4 \mathbf{P}_4,$$

where

$$\mathbf{P}_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Now we consider the matrix \mathbf{V}_8 . If we perform the following permutation of columns of this matrix:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 8 & 4 & 5 & 2 & 7 & 3 & 6 \end{pmatrix},$$

as a result we obtain the following matrix:

$$\bar{\mathbf{V}}_8 = \begin{bmatrix} 1 & -b & -b & b^2 & -b & b^2 & b^2 & -b^3 \\ b & 1 & -b^2 & -b & -b^2 & -b & b^3 & b^2 \\ b & -b^2 & 1 & -b & -b^2 & b^3 & -b & b^2 \\ b^2 & b & b & 1 & -b^3 & -b^2 & -b^2 & -b \\ b & -b^2 & -b^2 & b^3 & 1 & -b & -b & b^2 \\ b^2 & b & -b^3 & -b^2 & b & 1 & -b^2 & -b \\ b^2 & -b^3 & b & -b^2 & b & -b^2 & 1 & -b \\ b^3 & b^2 & b^2 & b & b^2 & b & b & 1 \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{V}}_4 & -b\bar{\mathbf{V}}_4 \\ b\bar{\mathbf{V}}_4 & \bar{\mathbf{V}}_4 \end{bmatrix}.$$

As previously, we can write:

$$\mathbf{V}_8 = \overline{\mathbf{V}}_8 \mathbf{P}_8,$$

where

$$\mathbf{P}_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

If we generalize the above considerations for $N = 2^n$ we can write:

$$\mathbf{V}_N = \overline{\mathbf{V}}_N \mathbf{P}_N \quad (17)$$

for $N = 2, 4, \dots, 2^n$. For $N = 2$ we can also write

$$\mathbf{V}_2 = \overline{\mathbf{V}}_2 \mathbf{P}_2 = \overline{\mathbf{V}}_2,$$

where \mathbf{P}_2 is an identity matrix of order two

$$\mathbf{P}_2 = \mathbf{I}_2.$$

The permutation matrix \mathbf{P}_N of order $N = 2^n$ can be obtained recursively from the permutation matrix $\mathbf{P}_{N/2}$ of order 2^{n-1} according to the following relation:

$$\mathbf{P}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{P}_N = \mathbf{S}_N \begin{bmatrix} \mathbf{P}_{\frac{N}{2}} & \mathbf{0}_{\frac{N}{2}} \\ \mathbf{0}_{\frac{N}{2}} & \mathbf{P}_{\frac{N}{2}} \mathbf{J}_{\frac{N}{2}} \end{bmatrix} \quad (18)$$

for $N = 4, 8, \dots, 2^n$. \mathbf{S}_N is the perfect shuffle permutation matrix of order 2^n , $\mathbf{J}_{N/2}$ is the counter-identity matrix of order $N/2$ and $\mathbf{0}_{N/2}$ is zero matrix. The perfect shuffle permutation is the permutation that splits the set consisting of an even number of elements into two piles and interleaves them. It can be written as follows:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & \dots & 2n \\ 1 & n & 2 & n+1 & \dots & 2n \end{pmatrix}.$$

For example

$$\mathbf{S}_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{J}_4 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

If we write the matrix \mathbf{V}_N as a product $\bar{\mathbf{V}}_N \mathbf{P}_N$ the expression (14) will take the form:

$$\mathbf{H}_N^a = \frac{1}{c^n} \bar{\mathbf{V}}_N \mathbf{P}_N \mathbf{A}_N^a \mathbf{P}_N^T \bar{\mathbf{V}}_N^T \quad (19)$$

The product $\mathbf{P}_N \mathbf{A}_N^a \mathbf{P}_N^T$ is a diagonal matrix, which has the same diagonal entries as the matrix \mathbf{A}_N^a but in different order and for a chosen parameter a it may be prepared in advance. If we denote this product multiplied by a factor $1/c^n$ by $\tilde{\mathbf{A}}_N^a$:

$$\tilde{\mathbf{A}}_N^a = \frac{1}{c^n} \mathbf{P}_N \mathbf{A}_N^a \mathbf{P}_N^T. \quad (20)$$

the DFRHT algorithm (16) will take the following form:

$$\mathbf{y}_N^{(a)} = \bar{\mathbf{V}}_N \tilde{\mathbf{A}}_N^a \bar{\mathbf{V}}_N^T \mathbf{x}_N \quad (21)$$

where the matrix $\bar{\mathbf{V}}_N$ can be generated recursively:

$$\bar{\mathbf{V}}_2 = \begin{bmatrix} 1 & -b \\ b & 1 \end{bmatrix} \quad \bar{\mathbf{V}}_{2k} = \begin{bmatrix} \bar{\mathbf{V}}_k & -b\bar{\mathbf{V}}_k \\ b\bar{\mathbf{V}}_k & \bar{\mathbf{V}}_k \end{bmatrix} \quad (22)$$

for $k = 2, 4, \dots, 2^{n-1}$.

3 Taking advantages of the particular structure of the matrix $\bar{\mathbf{V}}_N$

The most time-consuming operations by calculating the DFRHT transform according to (21) are multiplications of matrices $\bar{\mathbf{V}}_N^T$ and $\bar{\mathbf{V}}_N$ by the vector. Since in the matrix $\bar{\mathbf{V}}_N$ occur only following powers of b : $b^n, b^{n-1}, \dots, b^0 = 1$ we can write this matrix as follows:

$$\bar{\mathbf{V}}_N = \mathbf{A}_N^{(0)} + b\mathbf{A}_N^{(1)} + b^2\mathbf{A}_N^{(2)} + \dots + b^n\mathbf{A}_N^{(n)} \quad (23)$$

In the Figure 1 it was shown the way of calculating the matrix-vector product $\mathbf{y}_8 = \bar{\mathbf{V}}_8 \mathbf{x}_8$, using the expression (23). In this paper, the graph-structural models and data flow diagrams are oriented from left to right. Straight lines in the figures denote the operation of data transfer. We use the usual lines without arrows specifically so as not to clutter the picture. Note that the circles in this figure shows the operations of multiplication by a number inscribed inside a circle. In turn, the rectangles indicate the matrix-vector multiplications by matrices

Although it may seem strange, we will see that such an operation allows to reduce the number of multiplication and additions by multiplying the matrix $\bar{\mathbf{V}}_N$ by a vector. It should be noted that because of the recursive relation (22) between matrices $\bar{\mathbf{V}}_N$ and $\bar{\mathbf{V}}_{N/2}$, the following recursive relation between the matrices $\mathbf{A}_N^{(k)}$, $\mathbf{A}_{N/2}^{(k)}$ and $\mathbf{A}_{N/2}^{(k-1)}$ occurs:

$$\mathbf{A}_N^{(0)} = \begin{bmatrix} \mathbf{A}_{N/2}^{(0)} & \mathbf{0}_{N/2} \\ \mathbf{0}_{N/2} & \mathbf{A}_{N/2}^{(0)} \end{bmatrix} = \mathbf{I}_N$$

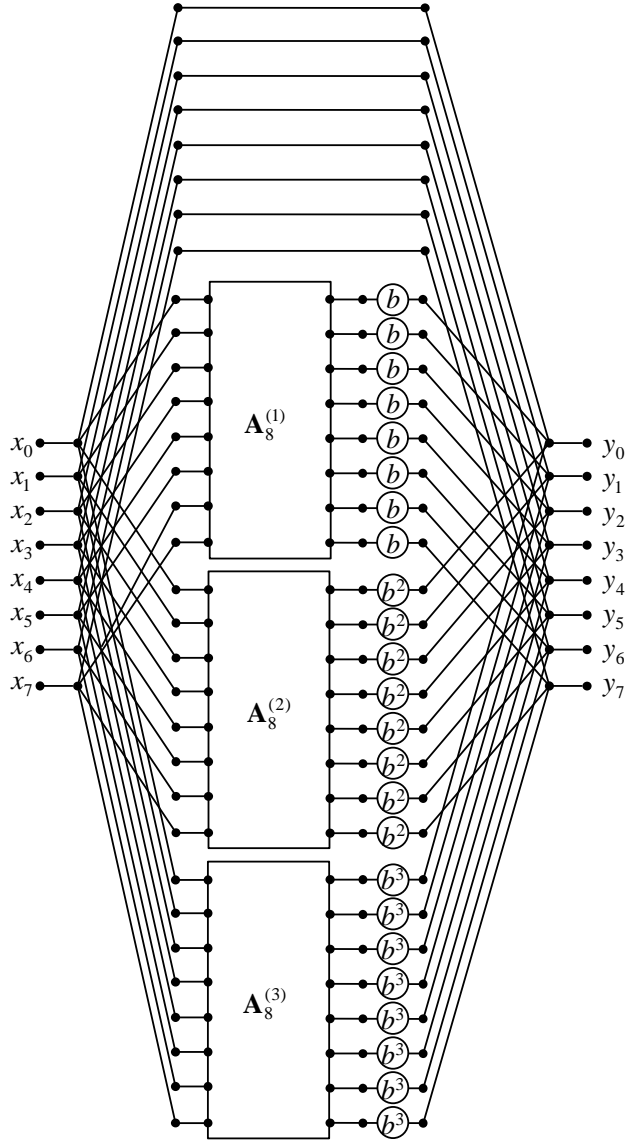


Fig. 1 The graph-structural model of calculating the product $\mathbf{y}_8 = \bar{\mathbf{V}}_8 \mathbf{x}_8$

$$\mathbf{A}_N^{(k)} = \begin{bmatrix} \mathbf{A}_{N/2}^{(k)} & -\mathbf{A}_{N/2}^{(k-1)} \\ \mathbf{A}_{N/2}^{(k-1)} & \mathbf{A}_{N/2}^{(k)} \end{bmatrix} \quad (24)$$

$$\mathbf{A}_N^{(n)} = \begin{bmatrix} \mathbf{0}_{N/2} & -\mathbf{A}_{N/2}^{(n-1)} \\ \mathbf{A}_{N/2}^{(n-1)} & \mathbf{0}_{N/2} \end{bmatrix}$$

for $k = 1, 2, \dots, n-1$ and $n = \log_2 N$, where

$$\mathbf{A}_2^{(0)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \mathbf{I}_2, \quad \mathbf{A}_2^{(1)} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}.$$

To clarify our idea we show the explicit form of expressions (23) and (24) for $N = 2$, $N = 4$ and $N = 8$ in an Example 3.

Example 3

$$\bar{\mathbf{V}}_2 = \mathbf{A}_2^{(0)} + b\mathbf{A}_2^{(1)}$$

where the matrices $\mathbf{A}_2^{(0)}$ and $\mathbf{A}_2^{(1)}$ are presented above.

$$\bar{\mathbf{V}}_4 = \mathbf{A}_4^{(0)} + b\mathbf{A}_4^{(1)} + b^2\mathbf{A}_4^{(2)}$$

where

$$\mathbf{A}_4^{(0)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_2^{(0)} & \mathbf{0}_2 \\ \mathbf{0}_2 & \mathbf{A}_2^{(0)} \end{bmatrix} = \mathbf{I}_4,$$

$$\mathbf{A}_4^{(1)} = \begin{bmatrix} 0 & -1 & -1 & 0 \\ 1 & 0 & 0 & -1 \\ 1 & 0 & 0 & -1 \\ 0 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_2^{(1)} & -\mathbf{A}_2^{(0)} \\ \mathbf{A}_2^{(0)} & \mathbf{A}_2^{(1)} \end{bmatrix},$$

$$\mathbf{A}_4^{(2)} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{0}_2 & -\mathbf{A}_2^{(1)} \\ \mathbf{A}_2^{(1)} & \mathbf{0}_2 \end{bmatrix}.$$

$$\bar{\mathbf{V}}_8 = \mathbf{A}_8^{(0)} + b\mathbf{A}_8^{(1)} + b^2\mathbf{A}_8^{(2)} + b^3\mathbf{A}_8^{(3)},$$

where

$$\mathbf{A}_8^{(0)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_4^{(0)} & \mathbf{0}_4 \\ \mathbf{0}_4 & \mathbf{A}_4^{(0)} \end{bmatrix} = \mathbf{I}_8,$$

$$\mathbf{A}_8^{(1)} = \begin{bmatrix} 0 & -1 & -1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_4^{(1)} & -\mathbf{A}_4^{(0)} \\ \mathbf{A}_4^{(0)} & \mathbf{A}_4^{(1)} \end{bmatrix},$$

$$\mathbf{A}_8^{(2)} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & -1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & -1 & -1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_4^{(2)} & -\mathbf{A}_4^{(1)} \\ \mathbf{A}_4^{(1)} & \mathbf{A}_4^{(2)} \end{bmatrix},$$

$$\mathbf{A}_8^{(3)} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{0}_4 & -\mathbf{A}_4^{(2)} \\ \mathbf{A}_4^{(2)} & \mathbf{0}_4 \end{bmatrix}.$$

Now we will evaluate the number of arithmetical operations, which are necessary to calculate the matrix-vector product $\mathbf{y}_N = \bar{\mathbf{V}}_N \mathbf{x}_N$. We note, that in a general case such an operation requires N^2 multiplications and $N(N-1)$ additions. Now we will calculate the numbers of multiplications and additions needed for this operation if we use the expression (23) for the matrix $\bar{\mathbf{V}}_N$, i.e.

$$\mathbf{y}_N = \mathbf{A}_N^{(0)} \mathbf{x}_N + b \mathbf{A}_N^{(1)} \mathbf{x}_N + b^2 \mathbf{A}_N^{(2)} \mathbf{x}_N + \dots + b^n \mathbf{A}_N^{(n)} \mathbf{x}_N \quad (25)$$

Since the non-zero entries of matrices $\mathbf{A}_N^{(0)}, \mathbf{A}_N^{(1)}, \dots, \mathbf{A}_N^{(n)}$ are only 1 and -1, no multiplications are needed when calculating the matrix-vector products $\mathbf{A}_N^{(k)} \mathbf{x}_N$. The only multiplications we have to perform are multiplications of the vectors $\mathbf{A}_N^{(k)} \mathbf{x}_N$ by the powers of b : $\mathbf{A}_N^{(1)} \mathbf{x}_N$ by b , $\mathbf{A}_N^{(2)} \mathbf{x}_N$ by $b^2, \dots, \mathbf{A}_N^{(n)} \mathbf{x}_N$ by b^n . Because the number b is constant and known, its powers b^2, b^3, \dots, b^n may be prepared in advance. Thus, the number of multiplication by calculating the matrix-vector product $\bar{\mathbf{V}}_N \mathbf{x}_N$ is equal to $nN = N \log N$. Let us examine the number of additions, we need to perform, when calculating the matrix-vector product $\bar{\mathbf{V}}_N \mathbf{x}_N$. The total number of additions consist of number of additions by calculating the matrix-vector products $\mathbf{A}_N^{(k)} \mathbf{x}_N$, and nN additions which are needed to calculate the sum of vectors: $\mathbf{A}_N^{(0)} \mathbf{x}_N, b \mathbf{A}_N^{(1)} \mathbf{x}_N, b^2 \mathbf{A}_N^{(2)} \mathbf{x}_N, \dots, b^n \mathbf{A}_N^{(n)} \mathbf{x}_N$. Since, according to (24), the matrices $\mathbf{A}_N^{(k)}$ have specific structures, the products $\mathbf{A}_N^{(k)} \mathbf{x}_N$ can be obtained by subtracting the products $\mathbf{A}_{N/2}^{(k)} \mathbf{x}_{N/2}^{(I)}, \mathbf{A}_{N/2}^{(k-1)} \mathbf{x}_{N/2}^{(II)}$ of twice smaller size and summing the products $\mathbf{A}_{N/2}^{(k-1)} \mathbf{x}_{N/2}^{(I)}, \mathbf{A}_{N/2}^{(k)} \mathbf{x}_{N/2}^{(II)}$ (excluding products $\mathbf{A}_N^{(0)} \mathbf{x}_N$ and $\mathbf{A}_N^{(n)} \mathbf{x}_N$ which can be obtained even in a simpler way). By $\mathbf{x}_{N/2}^{(I)} = [x_0, x_1, \dots, x_{N/2-1}]^T$ we denote the first half of the input vector \mathbf{x}_N and by $\mathbf{x}_{N/2}^{(II)} = [x_{N/2}, x_{N/2+1}, \dots, x_{N-1}]^T$ - the second half of this vector, as it was shown, for $N = 8$, in the Figure 2.

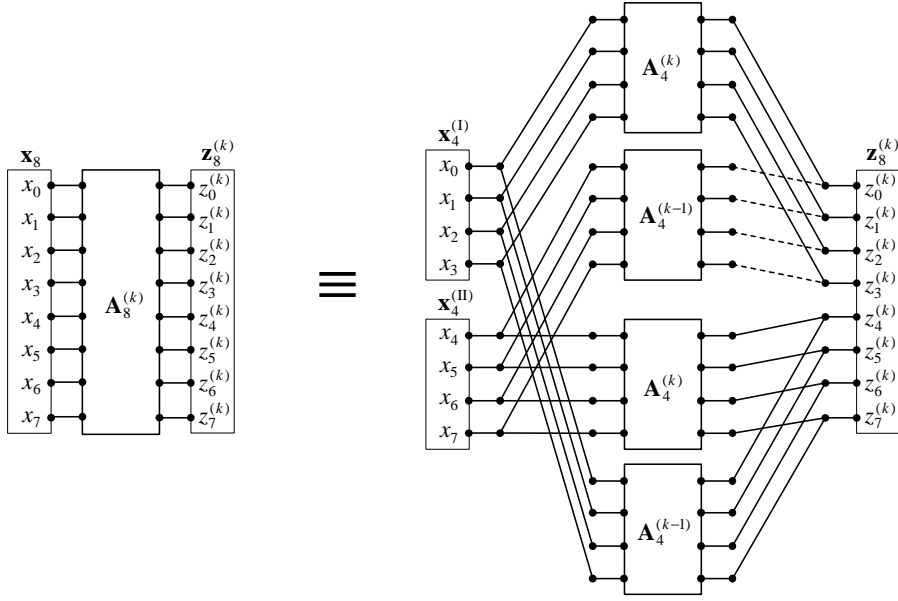


Fig. 2 The way of calculating the products $\mathbf{A}_8^{(k)} \mathbf{x}_8$ using the products of twice smaller size: $\mathbf{A}_4^{(k)} \mathbf{x}_4^{(I)}$, $\mathbf{A}_4^{(k-1)} \mathbf{x}_4^{(II)}$, $\mathbf{A}_4^{(k-1)} \mathbf{x}_4^{(I)}$, $\mathbf{A}_4^{(k)} \mathbf{x}_4^{(II)}$ for $k = 1, 2$

It should be noted that the products $\mathbf{A}_{N/2}^{(k)} \mathbf{x}_{N/2}^{(I)}$ and $\mathbf{A}_{N/2}^{(k)} \mathbf{x}_{N/2}^{(II)}$ are used to calculate both products $\mathbf{A}_N^{(k)} \mathbf{x}_N$ and $\mathbf{A}_N^{(k+1)} \mathbf{x}_N$. For example the products $\mathbf{A}_4^{(0)} \mathbf{x}_4^{(I)}$ and $\mathbf{A}_4^{(0)} \mathbf{x}_4^{(II)}$ are used to calculate $\mathbf{A}_8^{(0)} \mathbf{x}_8$ and $\mathbf{A}_8^{(1)} \mathbf{x}_8$. It allows to reduce the number of additions, because the some products are used twice. Of course, this procedure can be repeated and each of products $\mathbf{A}_{N/2}^{(k)} \mathbf{x}_{N/2}^{(I)}$, $\mathbf{A}_{N/2}^{(k-1)} \mathbf{x}_{N/2}^{(II)}$, $\mathbf{A}_{N/2}^{(k-1)} \mathbf{x}_{N/2}^{(I)}$, $\mathbf{A}_{N/2}^{(k)} \mathbf{x}_{N/2}^{(II)}$ can be calculated by summing (subtracting) products of twice smaller size and so on. It can be continued until calculating products of matrices $\mathbf{A}_2^{(0)}$ and $\mathbf{A}_2^{(1)}$ by two-element sub-vectors of the vector \mathbf{x}_N . The whole process of going down by calculating the product $\mathbf{y}_N = \bar{\mathbf{V}}_N \mathbf{x}_N$ is presented, for $N = 8$, in the Figure 3.

The expression (25) can be also written as the matrix-vector product, as follows:

$$\mathbf{y}_N = \bar{\mathbf{V}}_N \mathbf{x}_N = \mathbf{C}_{N \times (n+1)N} \mathbf{B}_{(n+1)N} \mathbf{A}_{(n+1)N \times N} \mathbf{x}_N \quad (26)$$

where

$$\mathbf{A}_{(n+1)N \times N} = \begin{bmatrix} \mathbf{A}_N^{(0)} \\ \mathbf{A}_N^{(1)} \\ \vdots \\ \mathbf{A}_N^{(n)} \end{bmatrix},$$

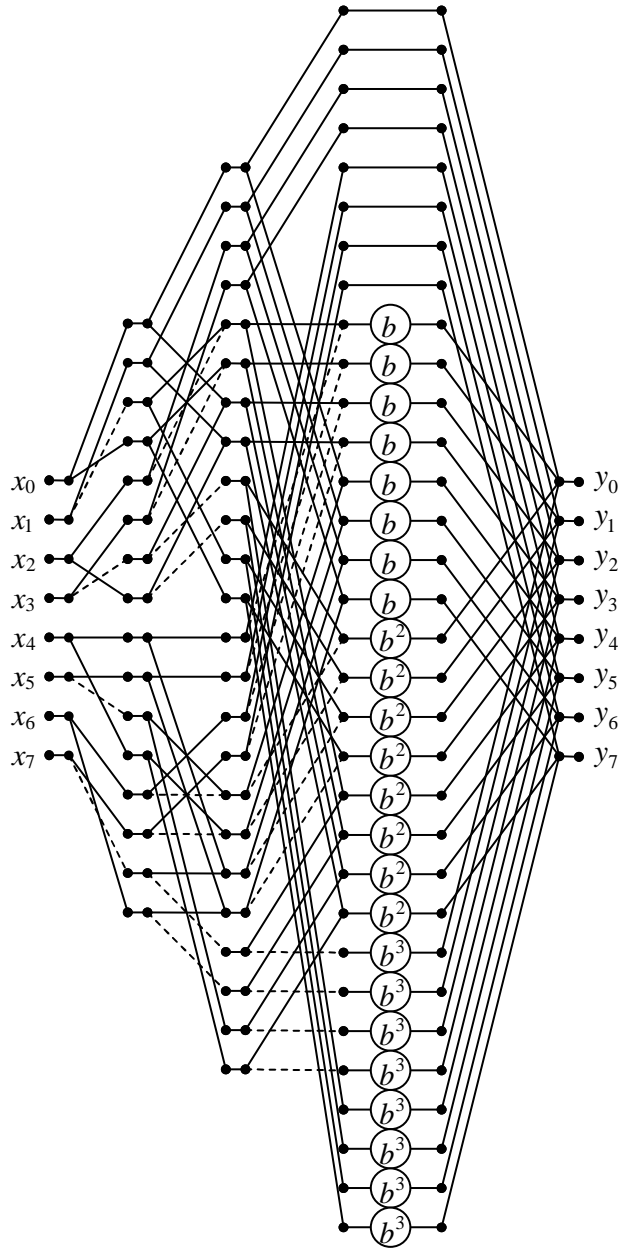


Fig. 3 Data flow diagram of calculating the products $\mathbf{y}_8 = \overline{\mathbf{V}}_8 \mathbf{x}_8$

$$\mathbf{B}_{(n+1)N} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & b & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & b^n \end{bmatrix} \otimes \mathbf{I}_N,$$

$$\mathbf{C}_{N \times (n+1)N} = \mathbf{1}_{1 \times (n+1)} \otimes \mathbf{I}_N.$$

The symbol \otimes denotes the Kronecker product of matrices, and $\mathbf{1}_{1 \times (n+1)}$ is the matrix (row vector) whose all entries are equal to 1. The matrix $\mathbf{A}_{(n+1)N \times N}$ is responsible for multiplications of the matrices $\mathbf{A}_N^{(0)}, \mathbf{A}_N^{(1)}, \dots, \mathbf{A}_N^{(n)}$ by the input vector \mathbf{x}_N , the matrix $\mathbf{B}_{(n+1)N}$ - for multiplications of those products by the proper powers of b , and the matrix $\mathbf{C}_{N \times nN}$ - for aggregation of results. The process of going down by calculating the product $\bar{\mathbf{V}}_N \mathbf{x}_N$, which has been presented in Figures 2 and 3, can be also described in the term of matrices product. It means factorisation of the matrix $\mathbf{A}_{(n+1)N \times N}$ into the product of n matrices

$$\mathbf{A}_{(n+1)N \times N} = \mathbf{A}_{(n+1)N \times nN} \mathbf{A}_{nN \times (n-1)N} \dots \mathbf{A}_{2N \times N} \quad (27)$$

The matrices which occur on the right side of expression (27) have the following forms:

$$\mathbf{A}_{2N \times N} = \mathbf{I}_{N/2} \otimes \begin{bmatrix} \bar{\mathbf{A}}_{2 \times 2}^{(0)} \\ \bar{\mathbf{A}}_{2 \times 2}^{(1)} \end{bmatrix} \quad (28)$$

where

$$\bar{\mathbf{A}}_{2 \times 2}^{(0)} = \mathbf{A}_2^{(0)} \otimes [1] \otimes \mathbf{I}_1 = \mathbf{A}_2^{(0)}, \quad \bar{\mathbf{A}}_{2 \times 2}^{(1)} = \mathbf{A}_2^{(1)} \otimes [1] \otimes \mathbf{I}_1 = \mathbf{A}_2^{(1)}.$$

$$\mathbf{A}_{3N \times 2N} = \mathbf{I}_{N/4} \otimes \begin{bmatrix} \bar{\mathbf{A}}_{4 \times 8}^{(0)} \\ \bar{\mathbf{A}}_{4 \times 8}^{(0)(2 \rightarrow)} + \bar{\mathbf{A}}_{4 \times 8}^{(1)(2 \leftarrow)} \\ \bar{\mathbf{A}}_{4 \times 8}^{(1)} \end{bmatrix}, \quad (29)$$

where

$$\bar{\mathbf{A}}_{4 \times 8}^{(0)} = \mathbf{A}_2^{(0)} \otimes [1 \ 0] \otimes \mathbf{I}_2, \quad \bar{\mathbf{A}}_{4 \times 8}^{(1)} = \mathbf{A}_2^{(1)} \otimes [0 \ 1] \otimes \mathbf{I}_2$$

and the matrix $\bar{\mathbf{A}}_{4 \times 8}^{(1)(2 \rightarrow)}$ denotes the matrix $\bar{\mathbf{A}}_{4 \times 8}^{(1)}$ which columns were circularly shifted by 2 positions to the right, and the matrix $\bar{\mathbf{A}}_{4 \times 8}^{(1)(2 \leftarrow)}$ denotes the matrix $\bar{\mathbf{A}}_{4 \times 8}^{(1)}$ which columns were circularly shifted by 2 positions to the left. The last matrix $\mathbf{A}_{(n+1)N \times nN}$ is defined as

$$\mathbf{A}_{(n+1)N \times nN} = \mathbf{I}_{N/N} \otimes \begin{bmatrix} \bar{\mathbf{A}}_{N \times nN}^{(0)} \\ \bar{\mathbf{A}}_{N \times nN}^{(0)(N/2 \rightarrow)} + \bar{\mathbf{A}}_{N \times nN}^{(1)((n-1)N/2 \leftarrow)} \\ \vdots \\ \bar{\mathbf{A}}_{N \times nN}^{(0)((n-1)N/2 \rightarrow)} + \bar{\mathbf{A}}_{N \times nN}^{(1)(N/2 \leftarrow)} \\ \bar{\mathbf{A}}_{N \times nN}^{(1)} \end{bmatrix} \quad (30)$$

where

$$\bar{\mathbf{A}}_{N \times nN}^{(0)} = \mathbf{A}_2^{(0)} \otimes [1 \ 0 \ \dots \ 0] \otimes \mathbf{I}_{N/2}, \quad \bar{\mathbf{A}}_{N \times nN}^{(1)} = \mathbf{A}_2^{(1)} \otimes [0 \ 0 \ \dots \ 1] \otimes \mathbf{I}_{N/2}.$$

Using the expression (27) the algorithm (26) of calculating the product $\bar{\mathbf{V}}_N \mathbf{x}_N$ can be written as follows:

$$\mathbf{y}_N = \mathbf{C}_{N \times (n+1)N} \mathbf{B}_{(n+1)N} \mathbf{A}_{(n+1)N \times nN} \mathbf{A}_{nN \times (n-1)N} \cdots \mathbf{A}_{2N \times N} \mathbf{x}_N \quad (31)$$

The expression (31) allows for evaluating the total number of additions, which are needed to calculate the matrix-vector product $\bar{\mathbf{V}}_N \mathbf{x}_N$. We assume that the input vector \mathbf{x}_N is real-valued. Each of matrices $\mathbf{A}_{(k+1)N \times kN}$, for $k = 1, 2, \dots, n$, is the direct sum of $N/2^k$ identical blocks and the single block is the vertical concatenation of $k + 1$ matrices. The first, indicated by $\bar{\mathbf{A}}_{2^k \times k2^k}^{(0)}$, and the last, indicated by $\bar{\mathbf{A}}_{2^k \times k2^k}^{(1)}$, do not need any additions or subtractions by multiplying them by a vector. The $k - 1$ others matrices, which are sums of $\bar{\mathbf{A}}_{2^k \times k2^k}^{(0)}$ and $\bar{\mathbf{A}}_{2^k \times k2^k}^{(1)}$, after circularly shifting their columns, so multiplying each of them by a vector needs 2^k additions. To calculate the product $\mathbf{A}_{(n+1)N \times nN} \mathbf{A}_{nN \times (n-1)N} \cdots \mathbf{A}_{2N \times N} \mathbf{x}_N$ we have to perform $\sum_{k=1}^n \frac{N}{2^k} (k - 1)2^k = Nn(n - 1)/2$ additions. The product of the matrix $\mathbf{B}_{(n+1)N}$ by a vector do not need any additions and the product of the matrix $\mathbf{C}_{N \times (n+1)N}$ by a vector needs nN additions. Thus the total number of additions by calculating the products $\bar{\mathbf{V}}_N \mathbf{x}_N$, according to (31), is equal to $Nn(n + 1)/2$.

Example 4 shows the explicit form of the algorithm (31) with all occurring in it matrices for $N = 8$.

Example 4 The algorithm (31) of calculating the product of matrix $\bar{\mathbf{V}}_8$ by the vector \mathbf{x}_8 is as follows:

$$\mathbf{y}_8 = \bar{\mathbf{V}}_8 \mathbf{x}_8 = \mathbf{C}_{8 \times 32} \mathbf{B}_{32} \mathbf{A}_{32 \times 24} \mathbf{A}_{24 \times 16} \mathbf{A}_{16 \times 8} \mathbf{x}_8,$$

where

$$\mathbf{A}_{16 \times 8} = \mathbf{I}_4 \otimes \begin{bmatrix} \bar{\mathbf{A}}_{2 \times 2}^{(0)} \\ \bar{\mathbf{A}}_{2 \times 2}^{(1)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix},$$

$$\mathbf{A}_{24 \times 16} = \mathbf{I}_2 \otimes \begin{bmatrix} \bar{\mathbf{A}}_{4 \times 8}^{(0)} \\ \bar{\mathbf{A}}_{4 \times 8}^{(0)(2 \rightarrow)} + \bar{\mathbf{A}}_{4 \times 8}^{(1)(2 \leftarrow)} \\ \bar{\mathbf{A}}_{4 \times 8}^{(1)} \end{bmatrix} =$$

$$\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0
\end{bmatrix},$$

$$\mathbf{A}_{32 \times 24} = \mathbf{I}_1 \otimes \begin{bmatrix} \overline{\mathbf{A}}_{8 \times 24}^{(0)} \\ \overline{\mathbf{A}}_{8 \times 24}^{(0)(4 \rightarrow)} + \overline{\mathbf{A}}_{8 \times 8}^{(1)(8 \leftarrow)} \\ \overline{\mathbf{A}}_{8 \times 24}^{(0)(8 \rightarrow)} + \overline{\mathbf{A}}_{8 \times 8}^{(1)(4 \leftarrow)} \\ \overline{\mathbf{A}}_{8 \times 24}^{(1)} \end{bmatrix} =$$

[illegible]

$$\mathbf{B}_{32} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & b^2 & 0 \\ 0 & 0 & 0 & b^3 \end{bmatrix} \otimes \mathbf{I}_8,$$

$$\mathbf{C}_{8 \times 32} = \mathbf{1}_{1 \times 4} \otimes \mathbf{I}_8.$$

4 The novel DFRHT algorithm

Now we return to the DFRHT algorithm (21). According to (23) the matrix $\overline{\mathbf{V}}_N$ can be written as the sum of the matrices $\mathbf{A}_N^{(0)}, \mathbf{A}_N^{(1)}, \dots, \mathbf{A}_N^{(n)}$ with co-

efficients $1, b, \dots, b^n$. The transposed matrix $\bar{\mathbf{V}}_N^T$ can be written as the sum of the transposed matrices $\mathbf{A}_N^{(0)T}, \mathbf{A}_N^{(1)T}, \dots, \mathbf{A}_N^{(n)T}$ with the same coefficients $1, b, \dots, b^n$:

$$\bar{\mathbf{V}}_N^T = \mathbf{A}_N^{(0)T} + b\mathbf{A}_N^{(1)T} + b^2\mathbf{A}_N^{(2)T} + \dots + b^n\mathbf{A}_N^{(n)T} \quad (32)$$

Since the matrices with the even indexes $\mathbf{A}_N^{(0)}, \mathbf{A}_N^{(2)}, \dots$ are symmetric and the matrices with the odd indexes are asymmetric the expression (32) can be written as follows:

$$\bar{\mathbf{V}}_N^T = \mathbf{A}_N^{(0)} - b\mathbf{A}_N^{(1)} + \dots + (-1)^n b^n \mathbf{A}_N^{(n)} \quad (33)$$

According to (26) the matrix $\bar{\mathbf{V}}_N$ from expression (23) can be transformed into the product

$$\bar{\mathbf{V}}_N = \mathbf{C}_{N \times (n+1)N} \mathbf{B}_{(n+1)N} \mathbf{A}_{(n+1)N \times N} \quad (34)$$

so the matrix $\bar{\mathbf{V}}_N^T$ may be also transformed from (33) into the following product:

$$\bar{\mathbf{V}}_N^T = \bar{\mathbf{C}}_{N \times (n+1)N} \mathbf{B}_{(n+1)N} \mathbf{A}_{(n+1)N \times N} \quad (35)$$

where the matrix $\bar{\mathbf{C}}_{N \times (n+1)N}$ is defined as follows:

$$\bar{\mathbf{C}}_{N \times (n+1)N} = \bar{\mathbf{I}}_{1 \times (n+1)} \otimes \mathbf{I}_N$$

and

$$\bar{\mathbf{I}}_{1 \times (n+1)} = [1 \quad -1 \quad \dots \quad (-1)^n].$$

The others matrices in the expression (35) are the same as that in the expression (26). Taking into account the decompositions (34) and (35) of matrices $\bar{\mathbf{V}}_N$ and $\bar{\mathbf{V}}_N^T$ respectively the DFRHT algorithm (21) will take the form:

$$\mathbf{y}_N^{(a)} = \mathbf{C}_{N \times (n+1)N} \mathbf{B}_{(n+1)N} \mathbf{A}_{(n+1)N \times N} \tilde{\mathbf{A}}_N^a \bar{\mathbf{C}}_{N \times (n+1)N} \mathbf{B}_{(n+1)N} \mathbf{A}_{(n+1)N \times N} \mathbf{x}_N \quad (36)$$

where the matrix $\mathbf{A}_{(n+1)N \times N}$ is decomposed according to (27). For example, for $N = 8$ this algorithm will take the following form:

$$\mathbf{y}_8^{(a)} = \mathbf{C}_{8 \times 32} \mathbf{B}_{32} \mathbf{A}_{32 \times 24} \mathbf{A}_{24 \times 16} \mathbf{A}_{16 \times 8} \tilde{\mathbf{A}}_8^a \bar{\mathbf{C}}_{8 \times 32} \mathbf{B}_{32} \mathbf{A}_{32 \times 24} \mathbf{A}_{24 \times 16} \mathbf{A}_{16 \times 8} \mathbf{x}_8.$$

Figure 4 shows a data flow diagram of the algorithm for 8 point DFRHT.

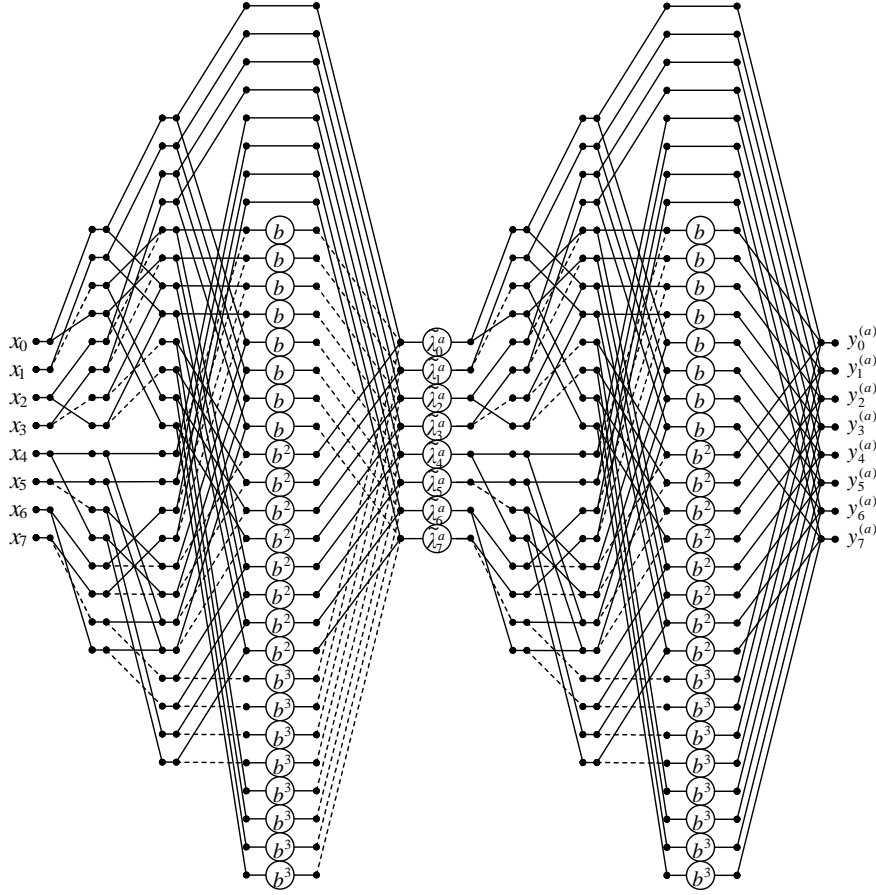


Fig. 4 Data flow diagram of the DFRHT algorithm (36) for $N = 8$

5 Discussion of computational complexity

Let us assess the computational complexity in term of numbers of multiplications and additions required for DFRHT calculation. Calculation of the discrete fractional Hadamard transform for a real-valued vector \mathbf{x}_N of length $N = 2^n$, assuming that the matrix \mathbf{H}_N^a defined by (4) is given, requires $N^2 = 2^{2n}$ multiplications of a complex number by a real number and $N(N-1) = 2^n(2^n-1)$ complex additions. Each multiplication of a complex number by a real number needs two real multiplications and each addition of two complex numbers requires two real additions. Hence the numbers of real multiplications and real additions required for computing the DFRHT using the naive method are equal to 2^{2n+1} and $2^{n+1}(2^n-1)$ respectively.

Let us now evaluate the computational complexity of the DFRHT realization with the help of the procedure (36). As it was discussed in the section 3, if we use the factorized representation of the matrices $\overline{\mathbf{V}}_N^T$ and $\overline{\mathbf{V}}_N$, calculat-

ing the product of the real-valued matrix $\bar{\mathbf{V}}_N^T$ and the real-valued vector \mathbf{x}_N requires nN real multiplications and $Nn(n+1)/2$ real additions. As a result, we again obtain the real-valued vector. Then there is computed the product of the complex-valued diagonal matrix $\tilde{\mathbf{A}}_N^a$ and the real-valued vector calculated previously (we assume that for a predetermined parameter a , the diagonal elements of this matrix were calculated in advance). The calculation of this product requires $2N$ real multiplications. The resulting complex-valued vector is then multiplied by the factorized matrix $\bar{\mathbf{V}}_N$. This operation requires $2Nn$ real multiplications and $Nn(n+1)$ real additions. The total numbers of arithmetic operations to compute DFRHT of size 2^n using our new algorithm are $N(3n+2)$ real multiplications and $3Nn(n+1)/2$ real additions. It is easy to check that even for small n the numbers of arithmetic operations required for realization of the proposed algorithm are several times less than in the naive method of computing.

Tables 1 and 2 display the numbers of multiplications and additions required for the DFRHT transform implementation of the real-valued input signal of the length $N = 2^n$. These numbers were calculated for three methods of the transform implementation: the direct multiplication of the DFRHT matrix by a vector of the input data, calculation according to authors' algorithm described in the work [11], and according to the algorithm (36) proposed in this article. It is easy to check that for $n > 5$ the number of arithmetic operations, required for DFRHT transform realization according to the proposed algorithm, is smaller than in the other two methods of DFRHT computing.

Table 1 Numbers of multiplications for mentioned algorithms

$N = 2^n$	direct method	method [11]	proposed algorithm
2	8	6	10
4	32	18	32
8	128	54	88
16	512	162	224
32	2048	486	544
64	8192	1458	1280
128	32768	4374	2944
256	131072	13122	6656
512	524288	39366	14848
1024	2097152	118098	32768

6 Summary

The article presents the novel algorithm for the DFRHT performing. The algorithm has a much lower computational complexity than the direct way of the DFRHT implementation. The computational procedure for DFRHT calculating is described in Kronecker product notation. The Kronecker product algebra is a very compact and simple mathematical formalism suitable

Table 2 Numbers of additions for mentioned algorithms

$N = 2^n$	direct method	method [11]	proposed algorithm
2	4	5	6
4	24	25	36
8	112	95	144
16	480	325	480
32	1984	1055	480
64	8064	3325	1440
128	32512	10295	4032
256	130560	31525	10752
512	523264	95855	69120
1024	2095104	290125	168960

for parallel realization. This notation enables us to represent adequately the space-time structures of an implemented computational process and directly maps these structures into the hardware realization space. For simplicity, we considered the synthesis of a fast algorithm for the DFRHT calculation for $N = 2^3$. However it is clear that the proposed procedure was developed for the arbitrary case when the order of the matrix is a power of two.

References

1. Pei S.C., Yeh M.H.: Improved discrete fractional Fourier transform. *Opt. Lett.* 22, 10471049 (1997)
2. Candan Ç.C., Kutay M.A., Ozaktas H.M.: The discrete fractional Fourier transform. *IEEE Trans. Sig. Proc.* 48, 1329-1337 (2000)
3. Pei S.C., Tseng C.C., Yeh M.H., Shyu J.J.: The discrete fractional Hartley and Fourier transforms. *IEEE Trans. Circuits Syst. Part II* 45, 665-675 (1998)
4. Pei S.C., Yeh M.H.: The discrete fractional cosine and sine transforms. *IEEE Trans. Sig. Proc.* 49, 1198-1207 (2001)
5. Yetik I.Ş., Kutay M.A., Ozaktas H.M.: Image representation and compression with the fractional Fourier transform. *Opt. Commun.* 197, 275-278 (2001)
6. Hennelly B., Sheridan J.T.: Fractional Fourier transform-based image encryption: phase retrieval algorithm. *Opt. Commun.* 226, 61-80 (2003)
7. Liu S., Mi Q., Zhu B.: Optical image encryption with multistage and multichannel fractional Fourier-domain filtering. *Opt. Lett.* 26, 1242-1244 (2001)
8. Nischchal N.K., Joseph J., Singh K.: Fully phase encryption using fractional Fourier transform. *Opt. Eng.* 42, 1583-1588 (2003)
9. Djurović I., Stanković S., Pitas I.: Digital watermarking in the fractional Fourier transformation domain. *J. Netw. Comput. Appl.* 24, 167-173 (2001)
10. Pei S.C., Yeh M.H., J.J. Shyu J.J.: Discrete fractional Hadamard transform. In: *Proc. IEEE Int. Symp. Circuits Syst.* 3, 179-182 (1999)
11. Majorkowska-Mech D., Cariow A.: An algorithm for discrete fractional Hadamard transform with reduced arithmetical complexity. *Przegląd Elektrotechniczny (Electrical Review)*. 88, 70-76 (2012)
12. Sylvester J.J.: Thoughts on inverse orthogonal matrices, simultaneous sign successions, and tessellated pavements in two or more colours, with applications to Newton's rule, ornamental tile-work, and the theory of numbers. *Philos. Mag.* 34, 461-475 (1867)
13. Korn G.A., Korn T.M.: *Mathematical handbook for scientists and engineers: definitions, theorems, and formulas for reference and review*. McGraw-Hill, New York (1968)
14. Yarlagadda R.K.R., Hershey J.E.: *Hadamard matrix analysis and synthesis*. Kluwer Academic Publishers, Boston (1997)

15. Yarlagadda R., Hershey J.: A note on the eigenvectors of Hadamard matrices of order 2^n . Linear Algebra Appl. 45, 43-53 (1982)